QL Today

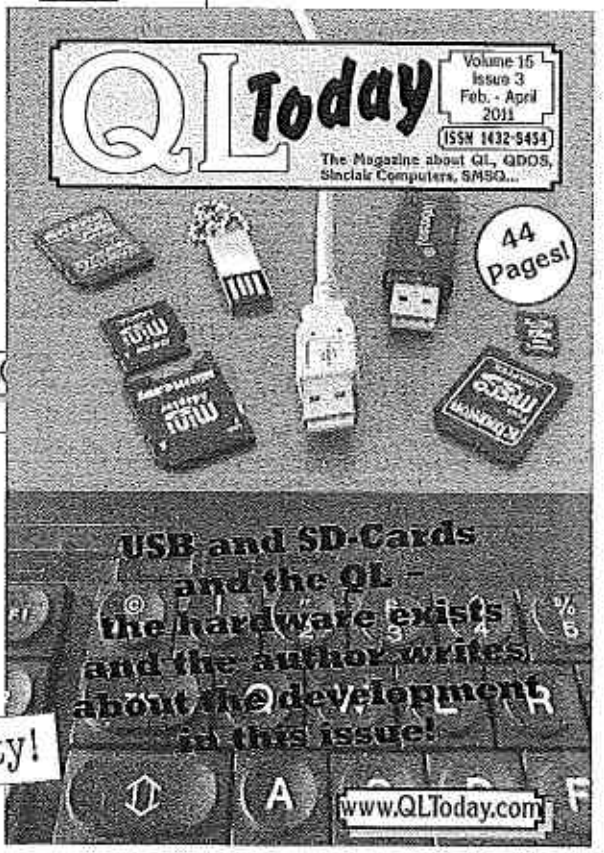
Volume 15
Issue 4
May - July
2011

ISSN 1432-5454

The Magazine about QL, QDOS, Sinclair Computers, SMSQ...

Vienna was great! How about a show in 2012? Where?

Half way there - next issue with DVD!



The previous covers of Volume 15 all show - it has been an interesting QL Today Year!

It's for all of us to turn the next Year into a Success too!

www.QLToday.com

Reality!

Contents

The deadline for the next issue is the 15th of August 2011

Advertisers

in alphabetical order

QL Today

ISSN 1432-5454

German office & Publisher:

Jochen Merz Software
Kaiser-Wilhelm-Str. 302
47169 Duisburg
Germany

Tel. +49 203 502011
Fax +49 203 502012
email: smsq@j-m-s.com
email: QLToday@j-m-s.com

Editor:

Geoff Wicks
Flat 5b
Wordsworth Avenue
Derby DE24 9HQ
United Kingdom

Tel. +44 1332 271366
email: gtwicks@btinternet.com
email: QLToday@j-m-s.com

Co-Editor & UK Office:

Bruce Nicholls
38 Derham Gardens
Upminster
Essex RM14 3HA
United Kingdom

Tel +44 20 71930539
Fax +44 870 0568755
email: qltoday@q-v-d.demon.co.uk
email: QLToday@j-m-s.com

QL *Today* is published four times a year, our volume begins on beginning of June. Please contact the German or English office for current subscription rates or visit our homepage www.QLTODAY.com.

We welcome your comments, suggestions and articles. YOU make **QL *Today*** possible. We are constantly changing and adjusting to meet your needs and requirements. Articles for publication should be on a 3.5" disk (DD or HD) or sent via Email. We prefer ASCII, Quill or text87 format. Pictures may be in _SCR format, we can also handle GIF or TIF or JPG. To enhance your article you may wish to include Saved Screen dumps. PLEASE send a hardcopy of all screens to be included. Don't forget to specify where in the text you would like the screen placed.

QL *Today* reserves the right to publish or not publish any material submitted. Under no circumstances will **QL *Today*** be held liable for any direct, indirect or consequential damage or loss arising out of the use and/or inability to use any of the material published in **QL *Today***. The opinions expressed herein are those of the authors and are not necessarily those of the publisher.

If you need more information about the UNZIP program which is used by our BOOT program to unpack the files, we suggest that you visit Dilwyn Jones' web site where you find more information about lots of interesting QDOS software and INFOZIP at http://www.dilwyn.uk6.net/arch/index.html

Editorial

by Geoff Wicks

The resilience of the QL community never ceases to amaze.

At the end of each QL Today volume I always pause to take a look at what we have achieved during the previous year. This time it was not so much the statistics that interested me, but the events we have reported.

In the first issue of volume 15 we devoted over 7 pages to a discussion that had taken place on the QL-users email group over the future of the QL. It was a wide ranging discussion during which we discovered that over half of UK QL-ers are neither QL Today readers, Quanta members nor subscribers to the QL-users list.

In issue 2 we reported on progress towards an electronic archive of QL Today which has now become a reality. All readers subscribing to volume 16 will receive a free copy.

In the same issue there was a brief mention of using an SD card on native hardware as a replacement for floppy disks. Issue 3 went a good deal further with reports of several hardware projects. Our cover story was on Adrian Ives' plan to produce a USB card and, in our news pages, we reported on Peter Graf's ideas for an SD card that could be fitted in the microdive slot. In this issue we report on work that Miguel Jodar has been doing.

We also report in this issue on a survey Dave Park conducted under 74 subscribers to the QL-users email group. About 20% of QL Today readers and 15% of Quanta members took part in the survey. Although most used an emulator, usually QPC2, as their main QL system two thirds still possessed some form of native hardware.

Quanta has played an important role in ensuring the continued viability of native hardware. Most recently it has supplied the financial backing for two batches of keyboard membranes.

Quanta has had a rough time during the past year with problems over its magazine, website, constitution and finance. Its future has been in doubt, but at the recent AGM some firm decisions were made to safeguard its survival. These include a move to a new web host with better technical support, a massive increase in the subscription and a proposed major revision of the constitution.

QL Today now moves into its 16th year. During the last 12 months we promised to produce a minimum of 120 pages. In practice we produced 170 and almost 86% of these were editorial, including more than 18 pages of news. Thirteen people contributed to the magazine.

It is the positive developments during the last year that have encouraged us to continue publishing, but we are aware that producing the magazine for another year will not be easy. In the last issue we mentioned the impending departure of Stephen Poole, although we still have a number of his articles on file. We have also completed publication of Tony Tebby's material that has played an important, and much appreciated, part in the last two volumes. Of some concern is a fall in the number of contributors. Thirteen writers compares with twenty one in the previous year and eighteen in the year before that.

The QL has a surprising resilience, but survival is only possible if sufficient QL-ers are prepared to get their hands dirty. We still need writers for QL Today.

SD and USB Card Progress

SD cards for native hardware have become a reality.

At the recent Quanta AGM Rich Mellor had one in use running games on a Trump Card QL.

Two weeks later Miguel Jodar posted a YouTube video showing a card in action:

http://www.youtube.com/watch?v=78xtsEKW4Lw

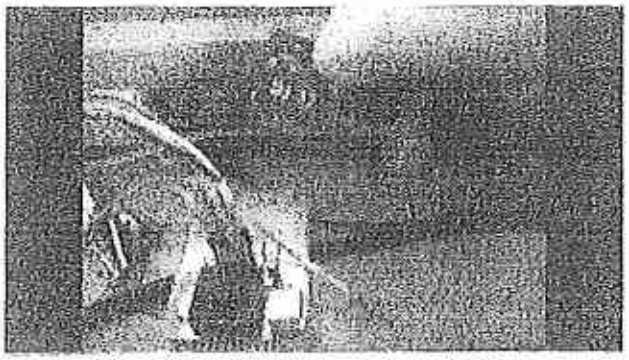
He had earlier posted a video of speed tests and comparison with a regular floppy drive:

http://www.youtube.com/watch?v=t0XvKy8wb-g

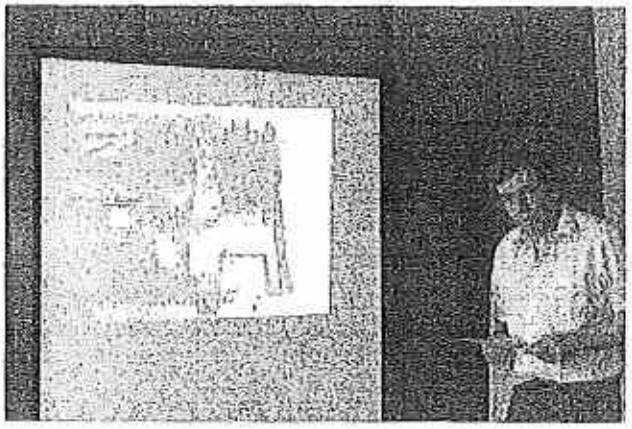
Miguel said he had been working on the project for two weeks and described the hardware as:

"An external ROM board, with EPROM for firmware (to be written), CPLD configured to act as an SPI interface, and a SD/MMC socket. Miscellaneous electronics include an integrated oscillator (50MHz at present), and a 3.3V LDO regulator."

Video streaming from SD card, thru QLSD interface



Also at the Quanta AGM Derek Stewart reported on progress in the development of Peter Graf's SD card. This will fit in the microdrive slot or alternatively can be used in the ROM slot.



Adrian Ives has also made progress on a USB card for native hardware, but the limitations of an unmodified QL have proved to be a major handicap. At the middle of April he announced he was withdrawing support for unmodified native hardware. However just over a week later, following feedback from several QL users, he reported he had found a solution for the problems, but added a warning that the maximum speed that could be achieved on unmodified hardware would be 4800 baud. However a QL fitted with a Hermes could achieve speeds of up to 19200 baud.

Adrian also reported that, although the project had been lengthy and costly, he had found little interest in it as a commercial product. Independently of Adrian Tony Firshman has taken an interest in the project and writes that, given the large amount of frustrating work Adrian has put into the project, he would like to see it go into production. *See late news page 33.*

QUANTA Changes

Quanta could face major changes during the coming 12 months.

The biggest is a 42% rise in the annual subscription from £14 to £20, although there will be a reduction for anyone opting for a three year subscription. The Quanta subscription has remained unchanged for over 20 years, but now no longer covers the organisation's costs. Last year subscription income covered only 67% of expenditure compared with over 77% ten years ago. In recent years the shortfall has been covered by the sale of second hand hardware, but this is a source of income that is rapidly drying up.

Quanta anticipates rising expenditure in the coming year with increases in website and magazine costs as well as additional committee

expenses. Until now Quanta has had free hosting of its website, but has decided to go over to a paid host costing about £300 a year in the hope that better technical support will help it over the problems reported in the last issue of QL Today.

Also planned is a major revision of the constitution following the problems that arose last year when no one could be found to replace John Gilpin as Treasurer. Although Quanta's constitution has been amended on 11 occasions during its 27 year life, it has never been given a complete revision, and still recognises only two QL systems, the black box and the Thor.

A provisional revised constitution was available at the recent AGM and it is planned to release this to the members in late summer for decision making on several, potentially controversial, matters. The new constitution makes provision for electronic participation in all meetings including the AGM. It is hoped that in this way more members, especially those living overseas, can participate in Quanta activities.

A full report of the Quanta AGM weekend appears elsewhere in this issue.

QL Survey

At the end of February Dave Park released results of a survey he had conducted under 74 subscribers to the QL users email group.

Half used an emulator as their main QL system and the majority of these were QPC users. 30% used various forms of native hardware and 9% a Q40 or Q60. About two thirds still had some form of native hardware. Only 3 users had just a single QL system, and 27 users (37%) had five or more systems.

28% were QDOS users, 12% Minerva and 57% SMSQ/E. 67% used a system with 4Mb or more and only 17% used one with less than 1Mb. 29% used their QL system daily and a further 32% at least once a week.

There was a high interest in buying new hardware or software with 84% expressing an interest in new hardware and up to 60% in new software. (These figures contradict the practical experiences of traders and developers.) Interest in new products fell off if these would cost more than 200 EUR.

95% of respondents could program in SuperBasic, but only 44% in assembly language.

35% were members of Quanta, 55% QL Today subscribers and 32% QL Forum users. 37% were former Quanta members and 24% former QL Today readers. The most popular website was that of Dilwyn Jones.

ZX81 Anniversary

For many QL-ers the ZX81, which celebrates its 30th Anniversary this year, was their introduction to programming. To mark the occasion Dilwyn Jones has released a CD of ZX81 emulators and programs.

The CD contains ZX81 emulators for both QL and PCs as well as for the Atari, Mac, Linux, Amiga OS4 and Spectrum computers. The disk also includes a Z80 emulator.

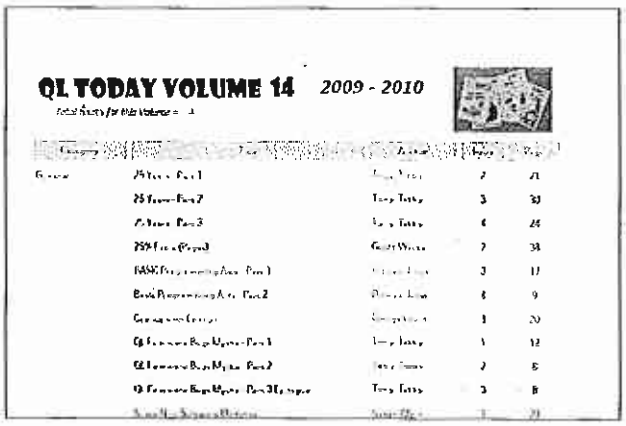
On the disk are numerous programs in .P and .81 formats and extensive documentation.

QL TODAY Indexes

Dilwyn Jones has also announced the release of indexes for QL Today volumes 12, 13 and 14.

The indexes have been compiled by Brian Kemmett, to whom QL Today is grateful. There are now indexes to every complete volume of QL Today.

These indexes can be downloaded free from http://www.dilwyn.me.uk/gen/qltoday/qltoday.html



Other DJ News

Dilwyn Jones writes:

"Thanks to Lee Privett, I have added a couple of useful items to my website.

1. A True Type font which looks like a QL font (based on the JS ROM font).

http://www.dilwyn.me.uk/fonts/index.html

2. Improved manual for the PCML disk interfaces, available as both PDF and Word .doc file (the latter is around 7MB in size, although the former is much smaller).

http://www.dilwyn.me.uk/docs/manuals/index.html"

GWASS Update

George Gwilt writes:

"Version 5.06 of GWASS is now available on my site **http:/gwiltprogs.info/**.

The new version allows much larger files to be assembled. There was previously a fault preventing an increase in the space allotted to user heaps.

There is as well a new version of the file explaining the 68K instruction set. This contains corrections to ADDX and SUBX. It also completes the entries for CALLM and RTM."

APOLOGY

George Gwilt and our readers deserve an apology from QL Today. The article "A Surprising Thing" in the last issue had already been published in QL Today. To compound our error we also spelt "surprising" incorrectly.

News

Programming in Assembler, Part 28
Application Sub-Windows - Continued

by Norman Dunbar

Introduction

Last issue I left you looking at a wonderful but practically useless program which displayed itself on screen and reacted only to the user clicking on a loose item or pressing the ESC key to quit the program. Apart from that, the most useful thing it did was to move the pointer into the middle of the application sub-window when the user pressed the TAB key.

This time, we get to add a little code, and see what happens when we hit an application sub-window. Let's get coding.

The Hit Routine

You don't actually need to have a hit routine for all your application sub-windows, there's nothing wrong with setting up an application sub-window in a program and then not having a hit routine. This is especially true if you intend to display information in it rather than handle user interactions and so on. As you will see when running the code below, the hit routine for an application sub-window gets called very frequently, so if you don't need one, don't use one.

You can disable the hit routine, if you don't need or want one, by setting the pointer to the hit routine to zero in the window definition file created by SETW. Now, it's time to get down to editing our new program.

We need to copy the two files we created last time, and rename them. So, copy 'ApplTest_asm' to 'ApplHitTest_asm', and 'ApplTestWin_asm' to 'ApplHitTestWin_asm'. These are going to be used in our first experiment.

First of all, we need to change one text object in the file 'ApplHitTestWin_asm', so edit that file and change the caption text to 'Application Window Hit Test' from 'Application Window Test 1'. Save the file.

Next up, we need to edit ApplHitTest_asm.

```
fname   dc.w  fname_e-fname-2
        dc.b  "Application Window Hit Test"
fname_e ds.b  0
```

Next we need to change the application sub-window hit routine. Look through the code for the routine named ahit0 and change it to the following.

```
; Application sub-window hit routine

ahit0   movem.l d1/d3/d5-d7/a0/a4,-(a7) ; Save the workers
        moveq #0,d1                     ; D1.W = Application sub window number
        moveq #0,d2                     ; D2.W = Ink colour = black
; A4 already set to Working definition
        jsr wm_swapp(a2)                ; Set channel id (A0) to the sub window

        movem.l a1-a2,-(a7)             ; A1 & A2 get corrupted
        lea hit,a1                      ; Text string to print
        move.w ut_mtext,a2              ; Print string vector
        jsr (a2)                        ; Print the message

        lea hitter,a1                   ; Hit counter location
        move.w (a1),d1                  ; Hit counter value
        addq.w #1,(a1)                  ; Increment counter
        move.w ut_mint,a2               ; Print integer vector
        jsr (a2)                        ; Print it

        movem.l (a7)+,a1-a2             ; Restore working registers

        movem.l (a7)+,d1/d3/d5-d7/a0/a4 ; Restore the workers
        moveq #0,d0                     ; No errors
        rts

; Strings and things go here

hitter  dc.w 0                          ; How many times have I been hit?

hit     dc.w hit_e-hit-2                ; Hit message
        dc.b 'HIT: '
hit_e   equ *
```

Then, right at the end, where we include our window definition, change the file name to suit our new name - from ApplTestWin_asm to ApplHitTestWin_asm.

```
; Pull in our window definition file.

        in  win1_source_ApplHitTestWin_asm
```

Save the file. We are done.

Note in the above code, the call to wm_swapp. This sets the channel id in A0 to point to the application sub-window that we specify in D1. The ink colour is set according to the value in D2.W and the sub-window is cleared. If we don't do this call, we might still print to the application window however, if we do, it's just luck! You should always ensure that the channel id in A0 has been explicitly pointed to the appropriate application window before attempting to print, cls, set paper or ink, etc when executing code within a hit routine. If you don't, any text printed by the code in the hit routine may well end up writing all over a loose item, or an information window or some random place in the window.
Obviously if you are running a program that has an application sub-window that doesn't have a hit routine, you will still need to call wm_swapp to make sure that the channel id in A0 points to where the sub-window is on screen - assuming of course, that you wish to write text to that sub-window as part of the application.

When you assemble it and execute it, note how the counter changes as you move the pointer over and around the application sub-window. It seems that you don't have to use the left mouse button or space bar to get a HIT. In fact using the right mouse button or ENTER both add 1 to the counter. The documentation says that "If there is no keystroke, or the keystroke is not the selection keystroke for a loose menu item or an application sub-window, then, if the pointer is within a sub-window, the hit routine is called, or else the loose menu item list is searched to find a new current item"

Press ESC to stop the program, then execute it again, try to keep the pointer outside the application sub-window. Now, press TAB. The pointer jumps into the sub-window, but what happens to the counter? It increases by two rather than one on every subsequent press of the TAB key.
This happens when you press some other key combinations, F1 increases the counter by two as well. Other letter or digit keys increment the counter by one.
I wonder why? Maybe, in the case of F1, the keystroke itself causes a hit and then the HELP event that the keystroke causes forces another hit? This doesn't explain the TAB key having the same effect though - that doesn't cause a hit. I mentioned above that a hit routine gets called very frequently didn't I?

OK, as an aside, I tested it. Using QMON2, I put a breakpoint at Ahit0 - the entry point for the application sub-window hit routine. On hitting the TAB key, I got a breakpoint. Looking at the registers I found that the pointer position in D1 was well outside my window limits, very strange. D2, the keystroke was set to -1 to indicate that an external keystroke fired the hit routine. There was no event in D4 - it was zero and D6, an undefined register was zero. So far so good, I noted down the registers and let the program continue.
It immediately stopped at the hit routine again, this time the pointer position had moved into my screen bounds from wherever it had been in hyperspace. D2 was now zero to indicate that no key has been pressed. D4 was still zero - so no events either. D6 had changed to $80. Wonder what that means?
Letting the program run again, I pressed F1 this time - without moving the pointer. Once again I hit the breakpoint. I could see the pointer position in D1 had not changed, D4 still showed no events, D2 showed no key press and D6 had returned to zero.
And again, I let the program run and it broke again. D6 was back at $80 again. D1, D2, D4 were all unchanged (as were all the other registers.)
I hit space this time, when I let QMON run the program. As expected this showed a $01 in D2 - the key press for a HIT is converted to $01, D6 was showing zero again. D4 still showed no events.
Once more, QMON let the program run and this time, I pressed ENTER. D4 showed the value $10 or the event number for a DO. The key press in D2 was set to $02 for a DO. D6 was zero and nothing else changed.
Finally, I pressed ESC to quit. Now, according to the docs, I should have stopped at Ahit0 again with D4 showing the CANCEL event, however, as documented above there was a loose item which had the ESC key set as the selection keystroke. That code was executed to exit from the program, rather than the application sub-window hit code being called with D4 set to the CANCEL event number.

The Advanced Hit Routine

So, that's our first very simple hit test program done and dusted. It's quite simple but quite useless, all it does is show you the running total of hits in the window. You will soon get bored of it.

For our next trick, we shall improve the utility to display full details of what data gets passed to the hit routine.
Copy 'ApplHitTest_asm' to 'ApplHitTest_2_asm', and 'ApplHitTestWin_asm' to 'ApplHitTestWin_2_asm'.

As before, we need to change one text object in the file 'ApplHitTestWin_2_asm', so edit that file and change the caption text to 'Application Window Hit Test 2' from 'Application Window Hit Test'. Save the file.

Next up, we need to edit 'ApplHitTest_2_asm'.

```
fname   dc.w  fname_e-fname-2
        dc.b  "Application Window Hit Test 2"
fname_e ds.b  0
```

Next we need to change the application sub-window hit routine. Look through the code for the routine named ahit0 and change it to the following.

```
; Application sub-window hit routine

ahit0   movem.l d1/d3/d5-d7/a0/a4,-(a7) ; Save the Hit Routine registers

        bsr.s apinit                    ; Initialise the sub-window
        bsr.s ptrpos                    ; Display details of the pointer position
        bsr.s keystr                    ; Display keystroke
        bsr events                      ; Print event details

        movem.l (a7)+,d1/d3/d5-d7/a0/a4 ; Restore the Hit Routine registers
        moveq #0,d0                     ; No errors
        rts
```

The main code in this advanced hit routine is simple - it stacks all the registers that we require to preserve throughout the hit routine, and makes calls to a few helper routines to carry out one specific task. I admit, this is not the most efficient method, but it allows me to split the code into manageable chunks for describing in the text.

```
; Helper - Initialise the sub-window.

apinit  movem.l d1-d2/a1-a2,-(a7) ; We need these registers later
        moveq #0,d1               ; D1.W = Application sub window number
        moveq #0,d2               ; D2.W = Ink colour
;A4 already set to Working definition
        jsr wm_swapp(a2)          ; Set channel id (A0) to the sub window
        movem.l (a7)+,d1-d2/a1-a2 ; Ptr position & keystroke back again
        rts
```

The first subroutine called simply initialises the application sub-window setting the ink to black and forcing the channel Id to cover the application sub-window. Any registers corrupted by the routine are stacked on entry and restored on exit.

```
; Helper - Display pointer position details.

ptrpos  movem.l d1-d3/a1,-(a7)    ; These get corrupted here
        lea ptrx,a1               ; 'Ptr_x: '
        bsr.s print               ; Print it

        move.l (a7),d1            ; Restore the old D1 again.

        swap d1                   ; Lo = pointer X, Hi = pointer Y
        bsr pr_int2               ; Print pointer X

        lea ptry,a1               ; ' Ptr_Y: '
        bsr.s print               ; Print it

        movem.l (a7)+,d1-d3/a1    ; Retrieve other registers
        bsr.s pr_int2             ; Print pointer Y
        rts
```

The code above preserves all registers that will be corrupted and then displays the current pointer position in absolute screen co-ordinates. These are relative to the 0,0 position of the entire screen and not relative to the 0,0 position of the actual main window for our application.

```
; Helper - Display keystroke

keystr  movem.l d1-d3/a1,-(a7)    ; These get corrupted here
        lea keystk,a1             ; 'Key: '
        bsr.s print               ; Print it

        move.l 4(a7),d2           ; Retrieve D2
        cmpi.b #-1,d2             ; External keystroke?
        bne.s k_hit               ; no, try a HIT

        lea keyext,a1             ; 'External'
        bra.s k_doit              ; Print & exit

k_hit   cmpi.b #1,d2              ; HIT?
        bne.s k_do                ; No, try a DO

        lea keyhit,a1             ; 'HIT'
        bra.s k_doit              ; Print & exit

k_do    cmpi.b #2,d2              ; DO?
        bne.s k_zero              ; No, must be a key code or zero

        lea keydo,a1              ; 'DO'
        bra.s k_doit              ; Print & exit

k_zero  cmpi.b #0,d2              ; Zero = no key pressed
        bne.s k_keys              ; Has to be a key press

        lea keyzero,a1            ; 'No key'
        bra.s k_doit              ; Print & exit

k_keys  move.w d2,d1              ; Need keystroke in D1.B
        moveq #io_sbyte,d0
        moveq #-1,d3
        trap #3                   ; Print keystroke
        bra.s k_done              ; Exit

k_doit  bsr.s print               ; Print message
k_done  movem.l (a7)+,d1-d3/a1    ; Restore working registers
        rts
```

The code above starts, as usual, by preserving the working registers. It then checks the value in D2 to see which, if any Key was pressed to cause a hit in the application sub-window. D2 can be any of the following:

- Negative 1 = the activation key was pressed to place the pointer into the application sub-window.
- 1 - HIT - the left mouse button was clicked within the application sub-window, or the space bar was pressed.
- 2 - DO - the right mouse button or the ENTER key was pressed while the pointer was within the application sub-window.
- Zero - no key or mouse button has been pressed.
- Anything else - this will be the upper cased key code for the actual key that was pressed.

If the TAB key is pressed, you might briefly see the "external keystroke" message flash across the screen quickly followed by "No key pressed" - as I mentioned previously, pressing TAB (the activation key for the sub-window) results in two separate calls to the hit routine.

```
; Helper - Print event details

events  movem.l d1-d3/a1,-(a7)     ; Save the usual bunch
        lea event,a1               ; 'Event: '
        bsr.s print                ; Print message
        move.w d4,d1               ; Event number
        bsr.s pr_int2              ; Print it
        movem.l (a7)+,d1-d3/a1     ; Restore the workers
        rts
```

Finally, we have the helper routine that displays details of whatever event was detected which caused the hit routine to be activated. As ever, the code starts by preserving the working registers and then examines D4 to see which, if any, event took place.

```
; Helper - Print string at (a1) to channel in A0. Then CLS to end of line.

print   move.w ut_mtext,a2         ; Vector to print string
        jsr (a2)                   ; Print it
        movem.l d1/d3/a1,-(a7)     ; These get corrupted
        moveq #sd_clrrt,d0         ; CLS to end of cursor line
        moveq #-1,d3
        trap #3                    ; Do it
        movem.l (a7)+,d1/d3/a1     ; Restore
        rts


; Helper - Print word int at (a1) to channel in A0.

pr_int  move.w (a1),d1             ; Get word to print
pr_int2 move.w ut_mint,a2          ; Print word int vector
        jsr (a2)                   ; Print it
        rts
```

The above routines are called by the main sub-routines themselves to display messages and numeric values on screen. The various messages are defined in the code below.

```
; Assorted TEXT messages etc follow.

ptrx    dc.w ptrx_e-ptrx-2
        dc.b 'Ptr_X: '
ptrx_e  equ *

ptry    dc.w ptry_e-ptry-2
        dc.b ' Ptr_Y: '
ptry_e  equ *

keystk  dc.w keystk_e-keystk-2
        dc.b $0a
        dc.b 'Key: '
keystk_e equ *

keyhit  dc.w keyhit_e-keyhit-2
        dc.b '1 = HIT'
keyhit_e equ *

keydo   dc.w keydo_e-keydo-2
        dc.b '2 = DO'
keydo_e equ *
```

```
keyext  dc.w keyext_e-keyext-2
        dc.b '-1 = External Keystroke'
keyext_e equ *

keyzero dc.w keyzero_e-keyzero-2
        dc.b '0 = No Key Pressed'
keyzero_e equ *

event   dc.w event_e-event-2
        dc.b $0a
        dc.b 'Event: '
event_e equ *
```

So, there you have it, a small and incredibly inefficient hit routine to display some of the data that are passed into an application sub-window hit routine when it is executed. I have not bothered to display the various window definition addresses etc - if you wish, feel free to create a 32 bit long to hexadecimal conversion routine to display these values.

I have deliberately left these out in an effort to save space in the magazine -- my listings can get a tad on the long side!

There is one final change we need to make to our source code, right at the end, where we include our window definition, change the file name to ApplHitTestWin_2_asm.

```
; Pull in our window definition file.

        in  win1_source_ApplHitTestWin_2_asm
```

Save the file. We are done.

When assembled and executed, the code in the hit routine displays a few details of register settings on entry to the hit routine. It's not very useful, but shows the pointer position in absolute screen co-ordinates as opposed to relative to the start of the actual sub-window (which would have been a lot more useful in my opinion), it shows the key press if any key was pressed and it shows which event, if any, occurred. Remember, only a limited number of events get through to a application sub-window hit routine.

If you press the TAB key and watch closely, you might see a brief message saying 'External Keystroke' before the text is replaced by 'No key pressed'. This shows, once again, that the TAB key results in two calls to the hit routine.

Conclusion

One thing has become obvious from even these two little routines, an application sub-window results in a huge number of hits! Even moving the pointer within a sub-window results in multiple hits. It might be better to display information - whatever the application needs to print on screen - to an information window instead. This should certainly save on processing time. However, as I mentioned above, only use a hit routine if you absolutely need one.

Failing this, the ideal hit routine for an application window should be hugely efficient -- and it should exit quickly when it doesn't need to do any [further] processing, rather than just doing everything each time the code is entered.

End of Part 28

Next time, we continue looking at application sub-windows, but we will be loading them with menus!

It's an Animal World

by Wolfgang Lenerz

It is always fun - and instructive - to read Tony Tebby's take on the QL and the computing world in general. I don't always agree with him, notably as concerns object oriented programming. I just thought about the example Tony has given, where he mentions a program that uses an "animal" object which has a "goFaster" method. Tony explains that, for flightless birds, once you use that method and the bird has to cross a river - well then the bird dies. He seems to conclude the object oriented programming (OOP) is bad.

I don't want to get into the OOP is good/bad discussion - it's mostly sterile: I've yet to see people being persuaded one way or another once they had formed their initial opinion. I do think, however, that the example chosen here is not a good example to demote the OOP concept. If you think about this example (I haven't been able to find it on the internet, unfortunately), it would stand to reason that the "animal" object not only has a "goFaster" method, but also a simple "go" method. What happens if you use this "go" method on a flightless bird and it gets to the river? Well, it will also drown (but perhaps more slowly?). Actually, the same fate would happen to a mouse, given a large enough river etc...

Clearly then, at first sight, this is a programming error (a bug) - the designer of the "animal" object just designed his object badly by not taking into account what would happen at a river crossing or if a tree was in the way etc... However if, because of such a programming errors, the programming language/concept in itself is bad per se, then I can only conclude that we're all, and always, using bad languages/concepts: I don't know about you, but I have yet to see any kind of programming language/concept that has not given rise to bugs - even those languages/concepts that use clearly defined data structures (ahem, buffer overflow, anyone?).

So, for me, this example isn't really something that shows that OOP is a bad thing. It just goes to show a bad program - at first glance.

I might even go a bit further, though: OOP is all about objects and their reuse - be it directly or through "inheritance". It is the latter concept that would be of interest to me here: it means that you derive one object from another: here, for example, you would create an object "mouse" that is based on the object "animal"[1], and you would only need to write the "mouse" specific parts of the object - all the other parts of the animal object would remain. And for the "go" method, you would have to make sure that the "going" part would only occur on a terrain the mouse could actually go on (what happens if it gets to a steel wall?). So, the "animal" object would just be a skeleton for the more precise objects.

Tony's argument is also that the application programmer, who uses the "animal" object and its "goFaster" method is doing exactly what the designer of the object provided for; but still gets a non functional program. Fair enough - any OOP based program is only as good as the objects it's based on. However, that only shifts the onus of correct programming from the application writer to the object writer. But, again, the same is true in "normal" programs - there are numerous examples where "well defined data structures" aren't - such as somebody writing things he shouldn't into the system variables etc. Or the well defined data structure isn't because the documentation gets is wrong - just as the "animal" object programmer documented his object's method wrong.

Ultimately, it means, as stated, that an OOP programmer will often have already existing objects which are then re-used in his program. True, the final program is only as good as the underlying objects. But isn't this true for non OO programs, too? Let's say you write in assembler or C. Don't you use libraries? Don't you use operating system calls etc.? And your program will only be as good as these underlying code parts, too....

What I like about OOP is that (at least to my mind) the potential for havoc is reduced. In a "normal" program, I have a well defined data structure which is used to exchange information/data - but there is nothing to stop me from misusing that data structure (by putting nonsense values in it) - intentionally or not. If the data is hidden from me, and I can only get at it via well-defined methods, that potential havoc I can create with it is reduced.

These are just my 2 cents worth on this debate. Also, please note that Tony is an IT professionnal, I'm not....

(1) Actually, you would probably create an object "mammal" based on "animal", then perhaps "rodent" based on mammal etc...

Challenging Year ahead!

by Geoff Wicks

On the first day of Quanta's AGM workshop, the telephone in the Gilpin household rang at 2.30 in the morning. Shortly afterwards Sarah Gilpin left Manchester to be with her father during his last hours. She asked John to remain in Manchester to be at the Quanta meeting. His day was to end after 11.00 p.m. by which time he was clearly exhausted.

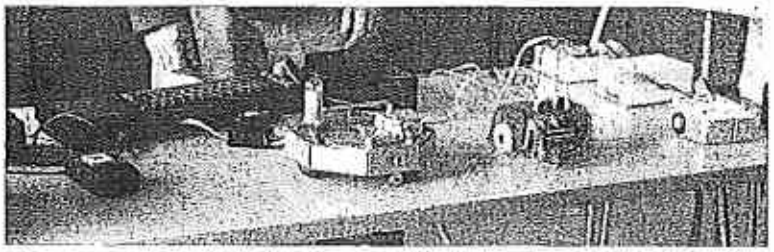
To add to the problems the AGM had been expected to be one of the most significant in years. Several important decisions were to be taken that could have implications for the future of Quanta. John was firmly told by several members that Sarah was not to rush back. "Family is more important".

Although I was booked to take part in a couple of activities, I had decided to take it easy for the rest of the show. I had already warned I would be arriving late and had planned not to attend the show dinner. I wanted to use the time instead on work to update the Just Words! website. In practice my weekend turned out to be very different from what I had planned.

About 20 people attended the show and on the Saturday there was a serious concern that Sunday's AGM would be inquorate, but no sense of panic. We would see what would happen the next day and, if necessary, decide what action to take then.

Three activities were planned for the Saturday afternoon and I had the honour of being the first to do a presentation using Quanta's new toy. (Quanta has invested in a projector to avoid having to beg or borrow one in future.) This was a live version of my article in the last QL Today on Roger Godley's Xchange software. At the moment the programs are only available from me, but I am hoping to place them on the Just Words! website if I am able to move it to a new host in the not too distant future. The delay is because I am working on an expansion of the help and advice page, the section that gets the most hits.

Next speaker was Dave Buckley, the QL's robot expert. This time he brought along table top models that could be programmed using turtle graphics. Dave has developed several robots over many years, some of which have become commercial products, and has used several different computers to steer them. As you would expect it has been easier to write the programs on Sinclair computers than on PCs.



Dave Buckley's robots or a future Quanta committee?

By now it was time to vacate the hall and there was no time for the third presentation.

Three hours later 14 of us met up for the traditional dinner in the Pond Quays Chinese restaurant. I had taken up one of the vacant places that had become available. The Manchester show dinner is very much an occasion for the "in-crowd" with the old jokes about vegetarian ducks and nicked napkins.

Ann Evans knitting covers to keep Dilwyn's QLs warm.

Just what is John Southern up to?

Sunday morning started with a talk, postponed from Saturday, by Rich Mellor on his wiki and software preservation projects.

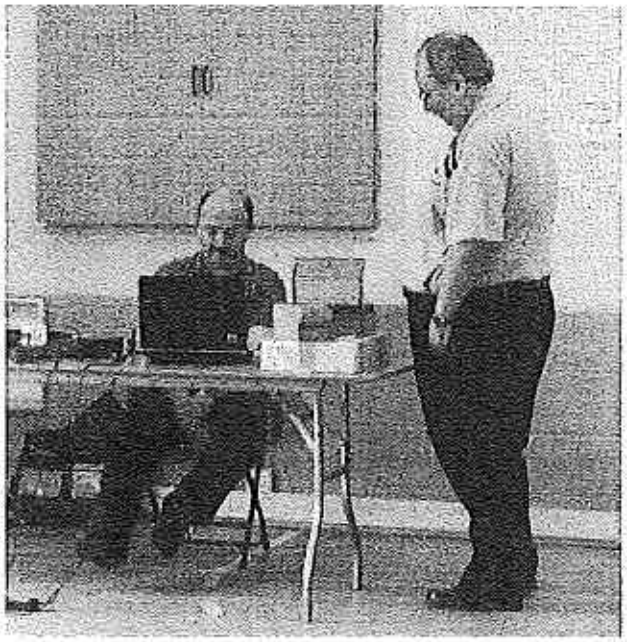
www.rwapadventures.com/ql_wiki

During the presentation Rich had as his guest Derek Stewart who described the latest developments on a card reader for the microdive or ROM slots. He illustrated it with a photo of a prototype in the microdrive slot.

Two informal discussions of Quanta business were planned for the Sunday morning. Both items were also on the agenda for the AGM but, given their importance, it was felt preliminary discussions to gauge the opinions of the members would save time and lead to more focussed decision making at the AGM.

Following the problems over the lack of a nomination for the office of treasurer last year and Quanta's breach of its constitution several members had suggested amending the constitution. Others had suggested certain sections needed rewording to remove ambiguities and make them easier to understand.

Although Quanta's constitution has been amended on 11 occasions over 27 years, it has never had a complete revision and is beginning to show its age. It recognises only two QL systems, the black box and the Thor, and, inevitably over 27 years, Quanta has adopted ways of working that are, strictly speaking, not constitutional. It is now proposed to give the constitution a complete spring clean to bring it more in line with Quanta's present procedures. A first draft was available at the meeting. With an eye to the future, provision is being made in the constitution for electronic participation in all meetings.



"If I ask nicely will he let me throw that PC out the window?"

Dilwyn guards an endangered species, a committee member from the South.

Quanta has never attempted to set up an internet link at its shows, although it does make extensive use of emailing between committee members to conduct its business. Other countries have used internet and video links at shows - most notably in Italy in 2008 when there was a presentation on QemuLator via a video link from Seattle. In spite of Quanta's lack of experience in this field secretary Alison Southern suggested that technological developments could make video-conferencing easier and welcomed the inclusion of the possibility in the draft constitution.

Rich Mellor, who was formerly a solicitor and company secretary, had prepared the draft. In a move unprecedented for Quanta, it is planned to release this to the members in late summer to allow a full discussion to take place before the final draft is produced at the end of the year. Some potentially controversial decisions have been deliberately left open for the members to make their decision. One of these is the six year rule under which, with a few exceptions, committee members have to step down after six years service.

The second discussion was on a proposal to raise the subscription. This has been unchanged for over 20 years but Quanta is now finding itself in increasing financial difficulties. Last year there was a deficit of £516 against an income of £2,635. The situation has been developing for some time but in recent years the potential shortfall of income against expenditure has been covered by the sale of second hand hardware via Rich Mellor. This is a source of income that is now drying up.

The formal AGM was held in the afternoon and I had been asked by the committee to chair the meeting in place of Sarah Gilpin. The annual reports from the officers were somewhat sober

with problems over the Quanta website, the constitution and finance, but on the positive side there appears to be some stability on the committee and a willingness to take firm decisions. Membership had held up well with a net loss of just 5 members in the last 12 months.

John Gilpin finally stepped down from the committee and his place as Treasurer was filled by Keith Dunbar. New to the committee is Lee Privett, who has recently returned to the QL scene. All the other committee members remained in office.

Quanta is expecting an increase in costs of the coming year. It hopes to solve the website problems by a move to a new host with better technical backup but costing approximately £300 a year. The new committee member comes from Essex and, although this is welcomed as a way of breaking the recent dominance of the north, it does mean committee expenses are likely to increase. There could also be extra postage and printing costs.

The members approved an increase in the subscription from £14 to £20, a rise of 42%. However there will be a reduced rate of subscription for people who pay three years in advance. They also approved the plans to produce a revised constitution in time for the 2012 AGM.

The next 12 months will be an uncertain time for Quanta. The rise in subscription could well result in the loss of members, but the attitude at the AGM was far from being defeatist. As one committee member put it Quanta is preparing for the future as it "starts to move towards 30 years of the QL by 2014".

Assembler Discussions

by George Gwilt and Norman Dunbar

Norman writes: *"Regarding George's letter, see the following which reproduces George's comments [GG] and is interspersed with mine [ND] in the finest tradition of these things!"*

[GG] I suppose the day may come when I have absolutely no comments on one of Norman Dunbar's Assembler articles - but so far that is still in the future.

[ND] Maybe one day this will happen, however, at the rate I seem to create errors and complete misunderstandings, I suspect it won't be any day soon! ;-)

[GG] I must first thank Norman for quietly correcting a fault occurring in the example EX0 which I give in EasyPEasy. On page 40, just before the trap #3 call to iop_flim, Norman sets register D2 to zero. In my example I did not do this.

[ND] I don't remember doing this explicitly to be honest, so I cannot comment here on exactly why I did it!

[GG] In fact, if D2 is not zero the call to iop_flim fails. I conclude from this that when my example EX0_BIN, which actually works, is run the contents of D2 are zero, presumably that being the initial value set when any program starts. It is unwise to bank on that though.

[GG] Norman has set out very clearly the steps to be taken to produce his window definition using SETW. Of course, as you might expect, I went through the process myself to see what happened. This showed up three things.

[GG] First, indication 17, headed Application Window 1, should have been three lines lower just before "Border size = 1". This leaves the three items, "Colour", "Xcsize" and "Ycsize" as part of "Object 1", as they should be.

[ND] This is indeed correct, I remember noting down everything I typed and the prompts on screen on a scrap piece of paper when I was creating the window, I obviously couldn't read my own writing when I transcribed it into the article.

[GG] Second, the colour of Object 1 should be black, not white.

[ND] Again, George is correct. If you look in the generated asm file for this window, the ink colour is indeed zero for black and not 7 for white.

[GG] Third, the size of the loose item in indication 19 should be 186x3 and not 183x3. An x-size of 183 cannot be set by SETW since it makes all x-sizes even, so this is, I guess, a simple misprint.

[ND] I assume George means the position as opposed to the size as the size is defined as 10 x 10. The position is itself 183 x 3 in the article, however, as George points out, SETW doesn't allow this setting and evens it up. Once again,

diving into the generated code file shows that the position is 186 x 3 as explained by George.

[GG] On page 42 Norman comments about op_con that D3 contains the timeout. In fact D3 here contains the code governing the type of open. This is relevant for opening files. In that case the types of open are 0 to 4 for OPEN, OPEN_IN, OPEN_NEW, OPEN_OVER and OPEN_DIR. To open an input pipe D3 has to contain the ID of the input pipe. However, for opening CON or SCR channels the value in D3 seems to be ignored. It is set in op_code to zero just to be on the safe side.

[ND] I obviously had a mental block here! I seem to think that all traps used D3 as the timeout when it is obviously not the case for IO_OPEN. As George correctly states, D3, in this case, is the open type parameter and not a timeout.

[GG] Finally, in the Note at the bottom of page 41, Norman explains the logic in the names "afun0_0" and "ahit0". I sense that he would have preferred that more meaningful names could have been given to these. I understand this.

[ND] No, I have no problems with the names chosen and fully understand them. My note at the bottom of page 42 was an attempt to explain why the names were as they were.

[GG] However, if there are several secondary main windows each with many loose items and, possibly several application windows as well, there really is a need to be able to pick out which routine belongs to what.

[ND] This is what I was attempting to explain in my note, that other windows would need different function names etc, and so the numbering system was a good way to do it.

[GG] To provide a clearer description the _asm file produced by SETW could be annotated by the programmer. Norman has in fact done this for the "Undefined Labels" in his example.

[ND] I got something right! ;-)

Dumping the QL

by Geoff Wicks

Two years ago I dumped the QL. It was a fascinating experience in which, paradoxically, the QL was never far away. If nothing else I learnt a lot about how we QL-ers think.

The story starts four years ago when my niece visited the town where my sisters and I grew up. She had fallen in love with a public house in one of the town's oldest buildings not knowing that there may have been a family connection. I had unsuccessfully researched legal documents in the county archives to determine whether or not the building had been the home or shop of my 3 times great-grandfather.

Sitting in the pub I foolishly promised her a CD of the family history - a promise that I failed to keep mainly because QL duties were taking up too much time.

Fast forward two years and I received an invitation to the first major family gathering in ten years. It was a now or never moment with a tight deadline. I had under three months to prepare the CD and that meant dumping the QL for the duration. Fortunately it was between issues of QL Today.

A CD of family history has many advantages over a paper version. It is light, cheap and easily burnt, but, above all, it can be interactive. Some members of the family would want a brief summary of the family history, but others would like more depth. An interactive CD, written in HTML code, would allow links to background documents for those people interested in details. For me it would be an interesting programming challenge as I had never written an interactive CD before. I soon discovered that without my QL knowledge and background I probably would not have succeeded.

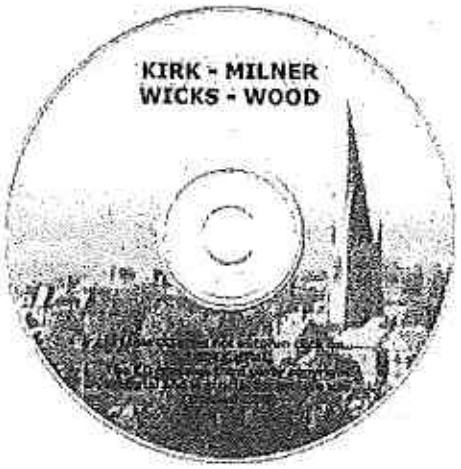
The first surprise was the realisation that the only times I had ever written HTML code was for the QL. The basics of HTML I had learnt from setting up the Just Words! website and later learnt some further design techniques from hosting pages on my site for the QL2004 and QLis21 shows. My program QL-2-PC Transfer can generate simple HTML code and I can still remember being wrapped on the knuckles by Marcel for some naughty shortcuts I made to transfer QL characters to their PC equivalents. My QL HTML experiences gave me the confidence to learn and try new techniques in spite of the tight deadline.

Importing the information into the HTML files went much smoother than I expected and once again the QL background helped. The simplicity of QL wordprocessors, spreadsheets and data-

bases has imposed a discipline upon QL users because we had to write such things as our own printer drivers. This means we are more familiar with the inner workings of these programs than people whose experience is more PC based and much more is done for them. This has given us more control over the way in which we use business programs. There are many Microsoft Word users who would not know what to do if you asked for an ASCII text file.

The importing was so quick that I was able to add more family tree information than I had originally intended. However I made a firm decision not to include detailed birth, marriage and death information as this would involve too much time consuming scanning work.

Scanning was the only area where the QL let me down. You cannot scan on a QL. I already had much of the background material in scanned form, but some First World War documents were technically difficult. Individual soldier records have suffered serious fire and water damage and the public are not allowed access to the originals. My grandfather's First World War record was of water damaged documents on A3 photocopies taken from a microfilm reader and in some cases then further reduced by photocopier to A4 before being scanned for the computer. You can imagine the quality problems this gave.



It all went a little too well for comfort and I had a first version of the CD prepared much earlier than expected. Then disaster struck. As soon as I tried the CD on a second computer it did not work.

It brought back vivid memories of my first pointer driven program. Before release I had tested this extensively on several systems and on arrival at the launch show I had proudly given Dilwyn a review copy for QL Today. With a commendable efficiency Dilwyn had immediately passed it on to Jim Hunkin to do the review. Half an hour later Jim announced, "It doesn't work!". I had fallen foul over some slight differences between SMS for the QXL and SMSQ for QPC which at the time I did not possess.

Once again I was glad to have my QL experience. Wise QL-ers do not panic when a program does not work, but instead start systematic and logical debugging. I soon discovered the problem was the old Microsoft one of its programs knowing better what you want to do than you do yourself. It was similar to the problem Dilwyn had had to solve to ensure his QL on a stick would run on all systems. Microsoft had used absolute addressing and I needed relative ones for all the files. Would I had solved this without QL programming and debugging experience? And as a QL-er I had few fears of checking and modifying the source code.

The problem was soon solved but then came another crisis that caused a further major delay, although this one was not technical.

I had returned to a local studies library to check the exact title of a history book from which I had quoted extensively. While there I came across a book of gruesome crimes that had been committed in the town and, almost perversely, had looked through it "just in case". Suddenly a name sprang out of the index. My three times great-grandfather had been a major prosecution witness to a brutal murder. Two butchers had had a quarrel over money. One had bludgeoned the other to death, dismembered the body and dumped the remains in a cesspit. The offence had been committed in a shop which had been part of the pub where the idea of the CD had been born.

Suddenly, with only a month to go, I found myself having to research and rewrite a substantial part of family history. Fortunately I was able to gain all the information I needed from newspaper archives in the local libraries and, from the reports of my three times great-grandfather's evidence, I was able to answer many questions about the family that had puzzled me for several years. The pub had, after all, not been the site of the family business or home.

When it finally came to production time I had the benefit of my Just Words! experience. Years of preparing new products and versions to tight deadlines for shows had taught me not to panic when things go wrong. I also had experience of producing disks and manuals and could estimate how long this would take. Yet another way in which the QL had helped.

In summary an interesting experience and a discovery that when you dump the QL it does not go away.

Queue Devices

by George Gwilt

In a previous article I described the vectors in the operating system allowing manipulation of queues. Here I examine how to make a queue into a device. Let us suppose that we want to be able to open a channel to a queue of a particular size so that we could put information into it for later extraction. We want to be able to type

```
OPEN#3,QUD_1024
```

for example.

It should now be possible to print a series of strings to #3 and later extract them, one by one, in the same order as they were entered. Furthermore, we want to have an error signalled either if the queue is empty when we want to extract something from it or if the queue is full when we try to add data to it.

How should this be accomplished?

Setting up a Serial Device

To set up a serial device we have to go through certain stages. We have to link in a device linkage block. This then becomes one of a chain of items starting at SV_DRLST in system variables. Our linkage block consists of four long words. The first will be set as a pointer to the next linkage block. The following three must contain the addresses of the three routines, I/O, OPEN and CLOSE. The I/O routines will be accessible by Trap #3 and the other two by Trap #2. Once the linkage block has been defined and set somewhere in ram we will use MT_LIOD to perform the linking.

Input/Output

When the device is opened to a channel the I/O routine is first entered by a Trap #3 call. If the operation is not completed at this first entry it is re-entered during each subsequent scheduler loop until it is complete.

On first entry to the I/O routine the operating system will have set the registers as follows.

```
D0      The operation code (as for Trap #3 with top 3 bytes set to zero)
D1      Parameter
D2      Parameter
D3      0
A0      Start of channel definition block
A1      Parameter
A2      Parameter
A3      Assumed start of driver definition
A6      System variables
A7      Supervisor stack
```

If the routine is re-entered during a scheduler loop D3 is set to 1.

For string operations D1 is set to zero. The string operations are IO_FSTRG, IO_FLINE, IO_SSTRG, FS_HEADS, FS_HEADR, FS_LOAD and FS_SAVE.

The I/O routine can use registers D2-D7 and A2-A6.

IO__SERIO

For comprehensive device drivers, such as CON and SCR, there may be many operations each of which requires coding. However, for a serial device driver the vector IO_SERIO provides a great simplification. Coding need be given for only three routines; IO_PEND, IO_FBYTE and IO_SBYTE. Given these, IO_SERIO will produce the code for the string operations listed above; ie the ones for which D1 is set to 0.

We will see later an actual example of the programming needed.

OPEN

The purpose of the OPEN routine is to set up a channel to the device.

As a minimum this requires checking the name of the device to see that it has been linked in and to find the values of any parameters and then allocating space for the channel block. In our case we need also to set up the queue and keep its address in a suitable place for the I/O routine.

The operating system sets the registers on entry to the OPEN routine as follows.

```
D3      Open type
A0      Pointer to device name
A3      Assumed start of driver definition block
A6      System variables
```

CLOSE

The space for the channel block has to be returned to the heap.

On entry the registers are set as follows.

```
A0      Channel block
A3      Assumed start of driver definition block
A6      System variables
```

The Queue Device – QUD

Before coming to the code which will create the device I must mention one particular problem. Each of the three vectors IO_QTEST, IO_QOUT and IO_QIN can signal the error "Not Complete" with the following meanings:

Vector	Meaning
IO_QTEST	Queue empty
IO_QOUT	Queue empty
IO_QIN	Queue full

The problem is this. When the error "Not Complete" is signalled in a Trap #3 routine, the operating system causes the routine to be re-entered during each subsequent scheduler loop until completion. In the case of the three vectors concerned, there will never be completion when the error "Not Complete" is signalled.

My simple solution is to replace the "Not Complete" error with "Not Found" for an empty queue and "Buffer Full" for a full queue. This is both more informative and stops the computer program hanging.

Coding for QUD

The coding to create the device QUD is now given.

```
**************************************************************
* QUD_ASM                                                    *
*                                                            *
* This device has one parameter. Thus                        *
*                                                            *
*         OPEN#3,QUD[_x]                                     *
*                                                            *
* will open a queue of size 256 or, optionally x.            *
*                                                            *
* (All types of open will work. Ie OPEN_IN, OPEN_NEW         *
* OPEN_OVER and even OPEN_DIR operate just like OPEN.)       *
*                                                            *
* To link in the device simply LRESPR the assembled file     *
*                                                            *
**************************************************************
*
* First set the addresses for IO_SERIO
*
```

```
          lea       ptrs,a0
          lea       io_ready,a2
          move.l    a2,(a0)+
          lea       fetch,a2
          move.l    a2,(a0)+
          lea       send,a2
          move.l    a2,(a0)

*
* Now set the addresses in the linkage block
* and link it in
*
          lea       linkage+4,a0
          lea       IO,a2
          move.l    a2,(a0)+
          lea       open,a2
          move.l    a2,(a0)+
          lea       close,a2
          move.l    a2,(a0)
          lea       linkage,a0
          moveq     #mt_liod,d0         link in QUD
          trap      #1
          moveq     #0,d0               Mark no error and . .
          rts                           . . return to BASIC

******************
* Linkage block *
******************
*
* The linkage block consists of 4 long words:
*
*         Link to next device
*         Pointer to IO routine
*         Pointer to OPEN routine
*         Pointer to CLOSE routine
*

linkage   dcb.l     4,0                 4 long zeros

*****************
* OPEN ROUTINE *
*****************
*
* This gets space from the heap for the 28 bytes needed for the
* channel block plus 16 + x bytes to hold the queue.
*
* The address of the queue is set in the long word $18 from the
* start of the channel block.
*
* The queue header starts $1C from the start of the channel block.
*
open      subq.l    #2,a7               space for 1 parameter
          movea.l   a7,a3               A3 -> parameter
          movea.w   io_name,a4
          jsr       (a4)
          bra.s     bad_exit  ----->    Not Found
          bra.s     bad_exit  ----->    Bad Parameter
          bra.s     ok
          dc.w      3
          dc.b      'QUD',0
          dc.w      1                   one parameter . .
          dc.w      ' _',256            . . default 256
ok
          moveq     #$2C,d1             get 44 bytes . .
          add.w     (a7),d1             . . plus queue size . .
          movea.w   mm_alchp,a4         . . for the channel block . .
          jsr       (a4)                . . and queue
```

```
          lea       $10(a0),a2          address of queue . .
          move.l    a2,$18(a0)          . . set in channel block
          move.w    (a7),d1             queue length . .
          ext.l     d1                  . . to D1.L
          movea.w   io_qset,a4
          jsr       (a4)                set the queue
bad_exit  addq.l    #2,a7               reset the stack
          rts

******************
* CLOSE ROUTINE *
******************
* At entry A0 points to the channel block

close     movea.w   mm_rechp,a4         return channel block . .
          jmp       (a4)                . . and queue to the heap

****************
* I/O ROUTINE *
****************
IO        movea.w   io_serio,a4         Deal with trap #3 . .
          jsr       (a4)                . . calls by IO.SERIO
*
* These three pointers are to:
*
*         io ready - here tests the queue
*         fetch a byte to D1
*         send a byte from D1
*
ptrs      dcb.l     3,0                 Space for IO_READY, FETCH and SEND
          rts

************************************
* The three routines for IO_SERIO *
************************************
*
* The error Not Complete is trapped and replaced by
* Not Found (-7) or Buffer Full (-5)
* These codes are set in D7 and later set in D0 if needed
*

fetch     movea.w   io_qout,a3
          bra       io_ready1

io_ready  movea.w   io_qtest,a3
io_ready1 moveq     #-7,d7              empty (not found)
          bra       send_1

send      movea.w   io_qin,a3
          moveq     #-5,d7              full (buffer full)
send_1    movea.l   $18(a0),a2          pointer to queue
          jsr       (a3)
          cmp.l     #-1,d0
          bne       send_2              no error
          move.l    d7,d0               set error -7 or -5
send_2    rts
```

Notes

1. Space is allocated for the channel block by the vector MM_ALCHP and released by MM_RECHP. These are atomic versions of the Trap #1 routines MM_ALCHP and MM_RECHP. These vectors must only be used in supervisor mode and must be used in the OPEN and CLOSE routines.

2. The name used to open the channel is analysed by IO_NAME. The call to IO_NAME must immediately be followed by three short branches. The first is taken if the device is not found, the second if the device is found but the parameters are faulty and the third if all is well.

I2C Interface for QL Emulators Part 1

by Ian Burkinshaw

One of the features that the Minerva II ROM provided for the original QL hardware was the I2C I/O (Input/Output) interface. TF Services also supported this interface with it's own interface products for both parallel and analogue applications.

As always with my projects this is to give you ideas and the basics. This is not a complete project in its own right with a real practical applications. I leave that up to you. However in another article I plan to show you what you can do with an old Sony radio to give it remote control. The techniques used can be used for other applications. More on this, another day. Also this is all at your own risk. But do have fun playing.

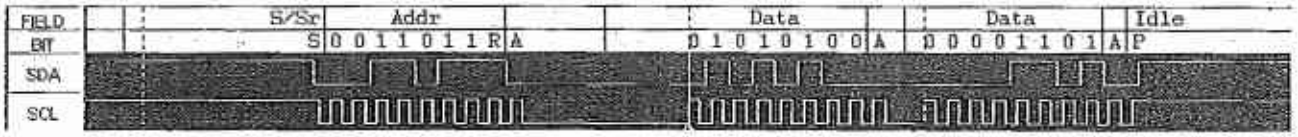
What I hope to show you in this short series is how to implement the use of the I2C interface using a QL Emulator such as QPC2. All the examples given have been tested on QPC2. But there is no reason why this should not work with other emulators. The only basic requirement is you need a USB port on your computer.

Some of you may still have the original TF Services I2C interfaces, if you have, then dig them out you can reuse these.

Now there is a way of recreating this interface using a product from ByVac. Their BV4221 product provides a USB to I2C interface. This device appears as a COM port just like a RS232 converter that I have used in my previous projects published in QL Today.

So back to the beginning, what is the I2C interface. This was an interface developed by Phillips. It is a two wire (plus ground) system. One wire is used as a clock (SCL) and the second (SDA) as a bi-directional data line. The computer, in our case a QL with a Minerva II ROM or a QL Emulator such as QPC2 with a ByVac BV4221 converter connected is the master station. This controls device(s) connected to it, which are called slaves. Each connected device has it's own address. So up to 254 devices can be connected to an I2C bus. Theses can be such things as bi-directional parallel ports, analogue to digital converters, digital to analogue converters, RTC (Real Time Clock), RAM (Random Access Memory) and digitally controlled potentiometers as examples. These are most likely, the most useful to us.

Here is an example of how these signals look, and gives a little insight to the protocol of the I2C system. More on this later.



The slave devices in most cases have up to 8 addresses available via 3 address connections in the chips themselves. Each type of device has it's own range of addresses. So using for example PCF8574 parallel I/O IC, you can use eight of these, in the address range 64 to 78 (Decimal), one device for each even address, I will come back to the odd numbers later. In fact you can have a further eight devices connected by using the PCF8574A device, which have the address range 112 to 126, again the even numbers. There are exceptions, for example the DS1307 RTC (Real Time Clock) device has only a single address. But it is unlikely that you would need more than one real time clock. However if you did there is nothing to stop you using two BV4221 converters on an emulator. Something you could not do with the original Minerva II QL solution.

Now as I have pointed out above the addresses I have quoted are even numbers. The reason for this is the 7 most significant bits are in fact the device address in the true term. The least significant bit is used to define reading or writing to a device. The rule being, even numbers are write addresses and odd numbers are read addresses. So for example if we take PCF8574 with all address pins held low, the write address would be 160 and the read address would be 161. See the address table. This shows all the addresses for all the devices featured in this series of articles.

QUANTA

Independent QL Users Group

World-wide Membership is by subscription only,
Offering the following benefits:
Bimonthly Magazine - up to 52 pages
Massive Software Library - All Free!
Free Helpline and Workshops
Regional Sub-Groups. One near you?
Advice on Software and Hardware problems
Subscription just £14 for Full Membership

PayPal (see QUANTA Web Site),
Cash, Cheques and Postal Orders Accepted

Now in our Twenty Eighth Year

Further details from the Membership Secretary

John Gilpin, 181, Urmston Lane,
Stretford, Manchester, M32 9EH (UK).
Tel. +44 (0) 161 865 2872
Email: membership@quanta.org.uk

Visit the QUANTA Web Site

Next QUANTA Sponsored Event

Annual General Meeting 2011 and Workshop

Date: Saturday/Sunday 16th/17th April 2011
Workshop from 12.00 Noon (Doors open 10 am for setting up) to 5.00 pm Saturday
And 9.00 am to 1.30 pm Sunday
Annual General Meeting 2.00 pm Prompt Sunday.

Venue: 3rd Davyhulme Scout Headquarters "The Endeavour", Conway Road, off Lostock Road, Davyhulme, Manchester. M41 0TF. Near M60 J9.

Full details from Chairman@quanta.org.uk

I2C Interface Addresses

Address Ranges	Read	Write
256x8 bit RAM (PCF8570)		
1010000	161	160
1010001	163	162
1010010	165	164
1010011	167	166
1010100	169	168
1010101	171	170
1010110	173	172
1010111	175	174
Parallel (PCF8574)		
0100000	65	64
0100001	67	66
0100010	69	68
0100011	71	70
0100100	73	72
0100101	74	74
0100110	75	76
0100111	76	78
Parallel (PCF8574A)		
0111000	113	112
0111001	115	114
0111010	117	116
0111011	119	118
0111100	121	120
0111101	123	122
0111110	125	124
0111111	127	126
A/D and D/A (PCF8591)		
1001000	145	144
1001001	147	146
1001010	149	148
1001011	151	150
1001100	153	152
1001101	155	154
1001110	157	156
1001111	159	158
RTC [Real Time Clock](DS1307)		
1101000	209	208
Digital Potentiometer (DS1803)		
0101000	81	80
0101001	83	82
0101010	85	84
0101011	87	86
0101100	89	88
0101101	91	90
0101110	93	92
0101111	95	94

So that I hope gives you a basic idea of what I2C interfacing is. So now how to implement this in practice.

I built myself a test board built on Vero Board, containing a PCF8574(A) parallel chip with LED's and switches to input and output to the chip. A PCF8591 AD and DA chip, a DS1307 RTC (Real Time Clock and finally a PCF8570 RAM chip. You do not have to build all of this if you don't wish to. For the purposes of the series of articles I have split this in to each chip in its own right.

I am going to start with the PCF8574(A) parallel chip.

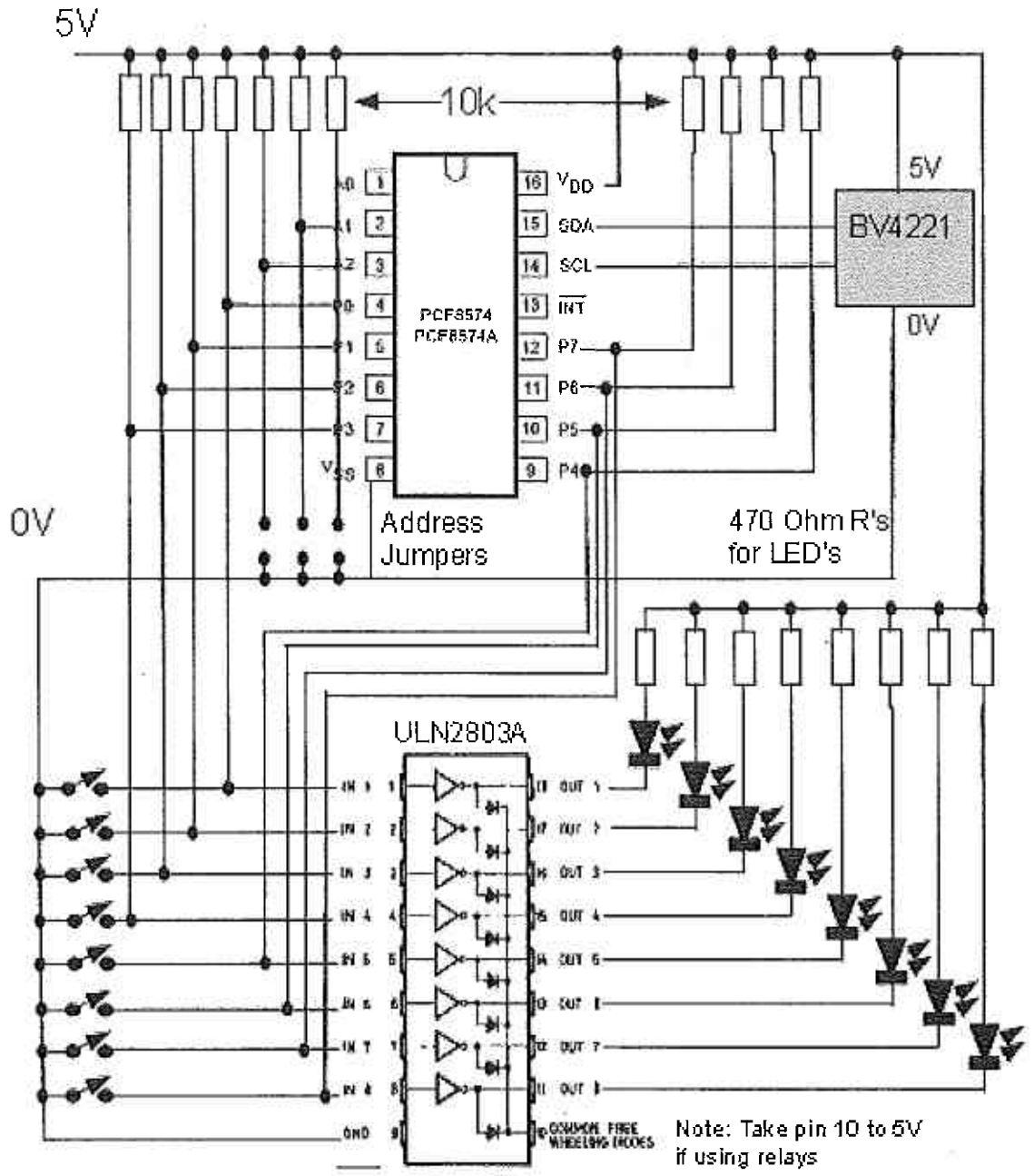
The circuit shown on the opposite page has the I2C parallel IC (PCF8574) shown at the top. This is the heart of this circuit the rest are the LEDs and a driver (ULN2803A) and switches to provide output indication and inputs. This is so you can fully experiment with this device. The reason for using the ULN2803A driver is two fold. First, the output from the PCF8574 could drive LEDs directly, but the LED and its resistor has to be from +5V supply to the port, this means when the output from the PCF8574 is high, that is a "1" the LED will not light. So it can get a bit confusing. The ULN2803A acts as an inverter, so that when the port is high, the LEDs light. The second point is the output current from the PCF8574 is limited, so if you want to drive relays for example you need a driver for these as well. The ULN2803A also contains the back EMF diodes so the device does not blow when the relay is released. If you are using relays please take pin 10 of the ULN2803A to +5V this connects all the output diodes together and provides the required protection.

All the ports are "pulled high" with the 10K resistors, so that the switches, when 'on', pull the inputs low. It is best to start with all the switches off. You have to ensure the PCF8574 outputs are high first. I will come back to this when we talk about the software.

The three address lines, A0, A1 and A2 are also "pulled" high as well. The jumpers pulling these low when required. You will see from the address table above this will make address for the PCF8574 '78' and for the PCF8574A '126'. If you wish to use more than one of each device type on a I2C bus then you need to fit jumpers as required. By using both PCF8574's and PCF8574A's you can have up to 128 I/O lines on a single I2C bus. That is 8 x PCF8574's providing 8 lines each and 8 x PCF8574A's again providing 8 lines each. That should be more than enough for most applications.

As to construction, stripboard is fine for this, just remember to cut the tracks as required on the reverse of the board. The photograph of my test board below should help you with the layout of this and the wire jumpers to build the circuit. Just follow the circuit diagram.

Circuit Diagram



Main Parts list

Part	Supplier	Part Number	Price each	Total
BV4221-V2	ByVac	BV4221-V2	£24.98	£24.98
PCF8574	RS Components	517-0687	£1.31 **	£6.55
PCF8574A	RS Components	517-0249	£1.63 **	£8.15
ULN2803A	RS Components	714-1167	£0.42 **	£2.10
8 x 470R Resistor	Maplin	M470R	25p	£2.00
11 x 10K Resistor	Maplin	M10K	25p	£2.50
8 x LED's	Maplin	CJ58N	54p	£4.32
Stripboard	Maplin	JP51F	£5.29	£5.29

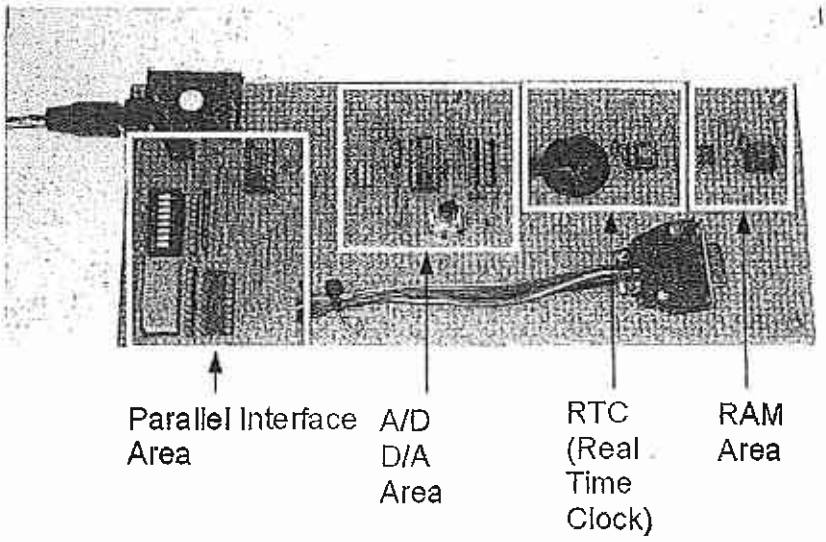
** Come in packs of five

You may wish to use IC sockets, which is not a bad idea, also you will need wire for the links etc. The BV4221-V2 is the most up to date version of the USB to I2C converter it is not the same size as the version I used. The prices are correct at time of writing, You may wish to find other suppliers that may be cheaper or do not have the minimum order quantities.

One of the nice features of this project is that the power comes from the USB port of your computer. However be careful not to load it too much. This simple test circuit is fine. But care needs to be taken if more than 100mA of current is required, this can cause problems and even damage to your computer. So if you think you will require more power such as powering relays for example, then using a stand alone PSU is a good idea. In this case, take the 5V wire from the BV4221 converter and connect your 5V PSU here.

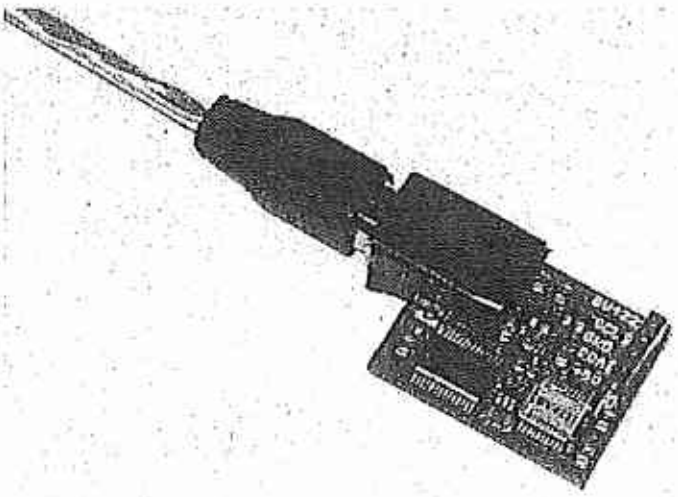
For those not so happy with soldering there is a simpler way, depending on your requirements. There is a range of pre built cards from ByVac, I have listed some of the options below, but do please look at the ByVac's web site for the most up to date information since they are constantly updating and changing their products. I must point out I have no connection with ByVac, I just use their products. There are others in the market. So worth a search on the internet.

Test card with labels



Now the software

```
5 CLS
10 BAUD 115200
15 parallel=126:REMark PCF8574A address, all address links open
17 adda=158:REMark PCF8591 address, all address links open
20 OPEN#3;ser2ir:REMark i=ignor hardware handshake, r=raw data
30 PRINT#3;CHR$(13);:REMark Carriage Return to set the baud rate in the USB to I2C converter,
   required on first pass to initialise USB to I2C converter.
40 print_reply
50 PRINT
60 PRINT#3;"V";CHR$(13);:REMark Command to USB to I2C coverter for firmware version.
65 PRINT "Return USB Converter Version Number:-";
70 print_reply:REMark Prints version number reply from USB to I2C converter
80 PRINT#3;"D";CHR$(13);:REMark Sets USB to I2C converter to receive decimal numbers, default is
   hex numbers.
85 PRINT "Decimal Mode Selected"
90 print_reply
100 ledflash
110 ledcount
120 input_test
200 PRINT "End":CLOSE#3:STOP
210 :
1000 DEFine PROCedure print_reply
1010 REPeat loop
1020 a$=INKEY$(#3)
1030 b$=a$
1040 PRINT b$;
1050 IF a$=">" THEN EXIT loop
1060 END REPeat loop
1070 END DEFine print_reply
1080 :
1100 DEFine PROCedure non_print_reply
1110 REPeat loop
```



```
1120 a$=INKEY$(#3)
1140 IF a$="›" THEN EXIT loop
1150 END REPeat loop
1160 END DEFine non_print_reply
1170 :
2000 DEFine PROCedure ledflash
2010 FOR a=1 TO 10
2020 PRINT#3;"s-";parallel;" 255 p";CHR$(13);:REMark s=start message to USB to I2C converter,
     p=end of message to USB to I2C converter.
2030 PAUSE 25
2040 PRINT#3;"s-";parallel;" 0 p";CHR$(13);
2050 PAUSE 25
2060 NEXT a
2070 END DEFine ledflash
2080 :
3000 DEFine PROCedure ledcount
3010 FOR a=0 TO 255
3015 PRINT a
3020 PRINT#3;"s-";parallel;" ";a;"p";CHR$(13);
3025 print_reply
3030 PAUSE 25
3040 NEXT a
3050 END DEFine ledcount
3060 :
4000 DEFine PROCedure input_test
4010 REPeat input_loop
4020 PRINT#3;"s-";parallel;" 255 p";CHR$(13);:REMark need to ensure any lines used as an input
     are set high.
4030 non_print_reply:REMark Stops printing the USB to I2C reply
4040 PRINT#3;"s-";parallel+1;" g p";CHR$(13);:REMark reads input data, note: the need to add 1
     to the address variable, this puts the I2C IS's into read data mode.
4050 print_reply:REMark prints both the address reply plus the data.
4060 IF INKEY$=" " THEN EXIT input_loop
4070 END REPeat input_loop
4080 END DEFine input_test
4090 :
```

The above routine tests the parallel port, both the output and input features.

So if all is OK, running this program will return the firmware version of the USB to I2C converter and then all the LED's should start to flash. They will flash 10 times at 0.5s interval's. Then there is a binary count from 0 to 255, this tests all permutations of the outputs. Then the final part of the program tests the input, just switch any or all of the switches 'on' and the numbers change on screen.

This simple routine shows how to use the protocol and how this works with the PCF8574(A) chip.

Line 10 set the baud rate of the port to be used. This can be anywhere in the range 9600 to 115,200 baud in there standard increments.

Line 30 sends CR (carriage return) to the USB to I2C converter. The converter sees this and it is this that sets the baud rate automatically on the converter itself.

Line 60 sends the character "V" followed by a CR, all commands to the converter require a CR at the end to tell the converter this string has ended.

Line 80 sets the converter into 'Decimal' mode, default is 'Hex' mode. I change it to decimal because it is easier to handle.

You will note that when the converter replies it returns the address of the first device it sees. So for example "126›". The reply routines are there to read these replies and stop them filling up the COM input buffer. Also the data from read commands, will be extracted from these replies as well.

You will see from the first diagram with the waveforms, that in the bit stream row the line start with an "s" - this is the start command. It is then followed by the address for the device we wish to control. If there is only one device then you need only send the "s" command. However I feel it is good practice to send the address of the device we wish to control, it saves rewriting thinks later if there are more devices added later.

So the start will look like "s-126 (The Device Address)...

Now we can add the data we wish to send to the device. In this case make all the output go 'high'. "s-126 255 p". So the "s" is the start as before followed by the device address, then the

number 255 which will set all the output 'High then "s" which is the "Stop Condition". Don't forget a CR at the end for it all to happen.

Now that sends commands to the device so how about reading data. This is where the odd address number comes in. Remember the address is the 7 MSB (Most Significant Bits) but there are 8 bits. The LSB (Least Significant Bit) is the one that controls the read/Write function of the I2C bus. So we just add "1" to the address to read data from the chip. Hence the address is 126 to write to the device and 127 to read the device. So the command now looks like this, "s-127 g p". So what does the "g" mean, I hear you ask. It is the command to 'Get' data from the device. So I have just given you some basic ideas of the concepts involved so do read the BV4221 pdf for more details on how to use the converter.

Next time we will look at the AD/DA, RTC and RAM chips and their use.

Some ByVac I2C products:

BV4502	I2C 2 Relay
BV4627-B	I2C 8 Relay
BV4236	I2C RTC and Temp
BV4237	I2C RTC, Temp and ADC
BV4506	I2C Keyboard Controller
BV700-B	Small Black keypad for use with the above keyboard controller
BV4240	I2C Bus Extender

Range of display controllers, to many to list here, so please see ByVac's web site www.byvac.com Then select 'shop', where you will find all the information on these products.
(Please note, The software with this article was not designed for these interfaces, so may need adapting)

References

http://www.byvac.com/bv3/index.php?route=product/product&product_id=88
(Please note, I used the original V1 of the BV4221, ByVac now supply V2 which also has a SPI interface. The commands are the same, so the programs listed in the article should still work.)

http://www.byvac.com/bv3/index.php?route=product/category&path=44

PCF8570 Ram Data Sheet
http://www.nxp.com/documents/data_sheet/PCF8570.pdf

PCF8574(A) Data Sheet
http://www.nxp.com/documents/data_sheet/PCF8574.pdf
http://focus.ti.com/lit/ds/symlink/pcf8574.pdf

PCF8591 Data Sheet
http://www.nxp.com/documents/data_sheet/PCF8591.pdf

DS1307 RTC (Real Time Clock)
http://datasheets.maxim-ic.com/en/ds/DS1307.pdf

DS1803 Digital Potentiometer Data Sheet
http://datasheets.maxim-ic.com/en/ds/DS1803.pdf

I2C Tutorials
http://www.robot-electronics.co.uk/acatalog/I2C_Tutorial.html
http://www.i2c.byvac.com/ar_foundation.php

TF Services I2C manual
http://www.dilwyn.me.uk/docs/manuals/index.html

Advanced I2C information, but still worth a read to understand I2C protocols
http://www.nxp.com/documents/user_manual/UM10204.pdf

Traverse

by Stephen Poole

This is an adaptation of an old board game for two players. White plays from left to right, while black plays from bottom to top. This requires a whole new chequerboard strategy.

To facilitate coding, I use letters in place of white pawns, versus numbers for black. Each player starts with a row of pawns on his edge of the board. White can move right, up or down. Black moves left, up or right. Each whole team of pawns must reach the opposing edge to win. There is no jumping or taking or doubling up on one square. The winner is the one who gets home first or blocks the moves of his opponent. This is why you must wait one round before being declared winner, as you may still be blocked!

The game took me about six hours to write and four to debug while waiting for the rain to cease during my summer holidays.

The difficulty levels determine the size of the board. The program might seem quite long, but this is done to simplify such problems as collision testing. The positions of all the labelled pawns are stored in the grid array G$(x,y), which is then searched systematically to determine the coordinates of the pawn being considered. Also the

empty squares are tagged with underscores, which can thereby easily be swapped with pawns. By using this tagging, it was much easier to debug step by step, as in this sort of program it is always easy to get the grid's 'across and down' indexes mixed up with the QL's 'row and column' addressing.

So start by RUNning and hit a key to say how many pawns you want. (For obvious reasons, the board will be one unit larger than your choice). Then you will see written on the screen which pawns are next to play. So hit either a number or letter for the pawn of your choice to move, then either 'u', 'd', 'l' or 'r' to move either up, down, left or right. If you make an impossible choice, the QL will burp, so try again. Otherwise it will beep gayly. If any one player cannot make a valid move, the other player wins, so then you must quit the game by hitting 'Q' when prompted.

I leave it to you to condense the code further, as there is plenty of scope for compression. But I forsee that such reductions will be at the expense of clarity, my intention as always being pedagogical.

So during my 'holiday QL sessions' I managed to recover some 14 programs from microtapes (and ED disks) which I have subsequently adapted for SMSQ/E as well as QDOS. This recovery gives me plenty of material for my 3D perspective animation project which I hope will be accepted for the Quanta Library, (as the routines contain DATA lists which are far too long to type in from magazines).

And there are plenty of subjects I look forward to tackling in the near future. Please feel free to modify the program as you wish, and happy coding...

```
100 ::
110 REMark Traverse_bas, by S.Poole, v6sept09
120  REMark beta-test by Bruno Coativy.
130 :
140 CLEAR: OPEN#1,con_16
150  WINDOW 512,256,0,0: CLS: PAPER 4: INK 7: CSIZE 3,1
160  PRINT 'Difficulty? 3 to 8': i$=INKEY$(#1,-1)
170  IF i$ INSTR '345678': CLS: ELSE GO TO 160
180  m=i$: n=m+1: ct1=0: ct2=0
190  DIM G$(n+1,n+1): : A$='12345678': B$='abcdefgh'
200 :
210 REMark prepare the board:
220  REMark Fill the board with under_scores:
230   INK 2: FOR ac=1 TO n: FOR dn=1 TO n: G$(ac,dn)='_': pr dn,ac,'_'
240  REMark Write the row of numbers:
250   INK 0: FOR ac=2 TO n: G$(ac,n)=A$(ac-1): pr dn,ac,A$(ac-1)
260  REMark write the column of letters:
270   INK 7: FOR dn=1 TO m: G$(1,dn)=B$(dn): pr dn,1,B$(dn)
280 :
290 REPeat game: player_1: player_2
300 :
310 DEFine PROCedure player_1
320  ik=0: INK ik: Choose_character 'NUMBER'
330  IF check_char(A$): get_coordinates: ELSE GO TO 320
340  IF get_direction('lur'): ELSE GO TO 320
350  IF bad_left : GO TO 320
360  IF bad_right: GO TO 320
370  IF bad_up   : GO TO 320: ELSE IF dn=1: IF test(1): won 1
380 END DEFine
390 :
400 DEFine PROCedure player_2
410  ik=7: INK ik: Choose_character 'LETTER'
420  IF check_char(B$): get_coordinates: ELSE GO TO 410
430  IF get_direction('urd'): ELSE GO TO 410
440  IF bad_up   : GO TO 410
450  IF bad_down : GO TO 410
460  IF bad_right: GO TO 410: ELSE IF ac=n: IF test(2): won 2
470 END DEFine
480 :
490 DEFine PROCedure won(wn)
500  IF wn=1: win$='Numbers': INK 0: ELSE INK 7: win$='Letters'
510  AT 0,0: CLS 3: BP 16: PRINT 'Bravo'!win$;'!': i$=INKEY$(#1,555)
```

```
520  AT 0,0: CLS 3: PRINT 'Another? y/n'!: i$=INKEY$(#1,300)
530  IF i$=='y': RUN: ELSE STOP
540 END DEFine
550 :
560 DEFine FuNction get_direction(d$)
570  pr 0,0,i$: pr 0,5,'?': j$=INKEY$(#1,-1)
580  IF j$ INSTR d$ : BEEP 12345,6: RETurn 1: ELSE : BP: RETurn 0
590 END DEFine
600 :
610 DEFine FuNction check_char(k$)
620 IF i$ INSTR k$: pr 0,0,i$: RETurn 1: ELSE RETurn 0
630 END DEFine
640 :
650 DEFine PROCedure get_coordinates
660  FOR ac=1 TO n: FOR dn=1 TO n: IF i$ = G$(ac,dn): RETurn
670 END DEFine
680 :
690 DEFine PROCedure Choose_character(ch$)
700 pr 0,8,ch$: pr 0,0,'?': i$=INKEY$(#1,-1): IF i$=='q': STOP
710 END DEFine
720 :
730 DEFine PROCedure write(t1,t2)
740  G$(ac,dn)='_': G$(t1,t2)=i$: INK 2: pr dn,ac,'_': INK ik: pr t2,t1,i$
750 END DEFine
760 :
770 DEFine FuNction bad_left
780 IF j$=='l': IF G$(ac-1,dn)<>'_' OR ac=1: BeP: RETurn 1: ELSE write ac-1,dn
790 RETurn 0: END DEFine
800 :
810 DEFine FuNction bad_up
820  IF j$=='u': IF G$(ac,dn-1)<>'_' OR dn=1: BeP: RETurn 1: ELSE write ac,dn-1
830 RETurn 0: END DEFine
840 :
850 DEFine FuNction bad_right
860  IF j$=='r': IF G$(ac+1,dn)<>'_' OR ac=n: BeP: RETurn 1: ELSE write ac+1,dn
870 RETurn 0: END DEFine
880 :
890 DEFine FuNction bad_down
900  IF j$=='d': IF G$(ac,dn+1)<>'_' OR dn=n: BeP: RETurn 1: ELSE write ac,dn+1
910 RETurn 0: END DEFine
920 :
930 DEFine PROCedure pr(p1,p2,m$)
940 AT p1,p2: PRINT m$
950 END DEFine
960 :
970 DEFine PROCedure BeP
980 BEEP 12345,67
990 END DEFine
1000 :
1010 DEFine FuNction test(chk)
1020 str$='': AA$='12345678'(1 TO m): ct1=0
1030 IF chk=1 THEN
1040    FOR f=1 TO n: str$=str$&G$(f,1)
1050    FOR f=1 TO n: IF AA$(f) INSTR str$: ct1=ct1+1
1060    IF ct1=m: RETurn 1
1070 END IF
1080 :
1090 BB$='abcdefgh'(1 TO m): ct2=0
1100 IF chk=2 THEN
1110    FOR f=1 TO n: str$=str$&G$(n,f)
1120    FOR f=1 TO m: IF BB$(f) INSTR str$: ct2=ct2+1
1130    IF ct2=m: RETurn 1
1140 END IF
1150 RETurn 0
1160 END DEFine
1170 ::
```

SER-USB Update

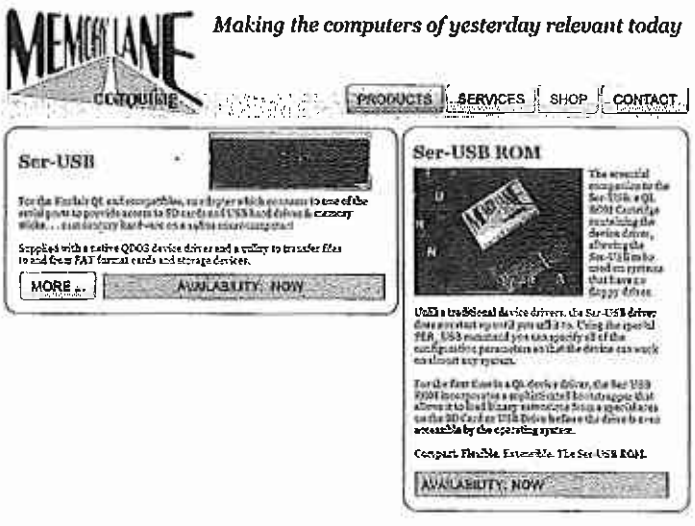
Adrian Ives is now selling his Ser-USB card. On his website he describes the card as:

"Measuring just 115x64x30mm the Ser-USB is compact yet powerful! Designed as a portable solution for moving data between different QL systems and emulators, the Ser-USB provides a bridge between the QL's simple serial hardware and the state-of-the-art "USBWiz" - an advanced circuit module that talks to modern storage devices; devices so sophisticated that they would have been the stuff of dreams when the QL was first conceived more than 25 years ago! Attach the Ser-USB to one of the QL's serial ports, load the device driver (which is available on ROM cartridge for supreme convenience) and you will have access to SD Cards and USB storage devices, all of them formatted as native QDOS volumes.

The Ser-USB is supplied with a 5V multi-voltage switching power supply with interchangeable plug tops for UK, Europe, USA and Australia. It also comes with one serial lead of your choosing: 8 pin Mini-DIN to QL SER1, QL SER2, or 9 pin D-SUB DCE and DTE variations. If your connection is not one of these, just ask and we can make a custom lead. A printed user manual is supplied, along with the device driver on QL format floppy disk, or, if you prefer, on a PC CD-ROM.

The Ser-USB is not intended to be a replacement for a conventional fixed disk system. Its primary purpose is for use as a device to enable the transfer of files between many different QL configurations using QDOS-formatted media. Its performance is limited by the performance of the serial ports of the machine to which it is connected. On a standard QL, this means that the Ser-USB is limited to 4800 baud. On a QL with Hermes, that is raised to 19200 baud, and with superHermes to 57600. On Q40/Q60 and PC/Mac emulators, it will be limited by the maximum speed of the serial port hardware. The maximum speed of the Ser-USB device is 230Kbaud, but it may not be possible to achieve this due to other system limitations (such as processor speed) and you should not anticipate being able to use speeds higher than 115Kbaud reliably. Please note that, at the time of writing, the Ser-USB requires extended memory to run on a standard QL. Also, without a ROM cartridge, you will require a floppy drive connected to the QL in order to load the Ser-USB driver software. Although the software has been through a long period of testing, it is the first new device driver for the Sinclair QL in many years and there are bound to be some teething problems as it is deployed to different environments. We have a policy of continuing support, development and optimisation of the driver, and will normally provide fixes to identified problems by means of a software download. However, if we are unable to resolve an issue, and you cannot use the Ser-USB in your configuration, we do offer a full money back guarantee within 14 days from the date of purchase. Updates to the base version 1.x driver are free (by download from our web site) for the lifetime of the product. Utility software updates (again, by download) are provided free of charge for one year from the date of purchase. If you purchase the ROM cartridge, the price includes the cost of two EPROM re-burns, but not the postage and packing to return the ROM cartridge. Finally, although it uses a custom manufactured printed circuit board, each and every Ser-USB is hand-made. This means that there may be some slight differences between the units. Also, the finish that it is possible to obtain by hand crafting the enclosure will not be as good as that obtained by dedicated machine tooling, and the slots and socket openings may show some small imperfections"

The card costs £125 plus post and packing (UK: £10 - EU: £15 - USA: £21). Payment can be by Paypal or UK bank cheque.



Letter-Box

Per Witte writes: Some comments on George Gwilt's "A Surprising Thing"

You may not be aware (or more likely, you have forgotten!) that virtually any job can be made into an EXecutable Thing. You simply use one of the Thing keywords, like HOT_RES or HOT_RES1 to load it. I have just tried it in QPC2 with net_peek_bin version 3.34

```
ERT HOT_RES1("n", "‹source directory›net_peek_bin")
```

Then pressed `ALT+'n'` to invoke, or if already running, to PICK.

I have always interpreted the manual (QL Reference Manual) to mean that a SEPARATE block of memory is required for the Thing linkage block. To quote from QRM section 17, page 9: "Two areas are allocated, one for the Thing contents and the other for the Thing linkage. The contents may already be present in RAM or ROM/EPROM, but the linkage has to be in RAM."

First your program has to reserve space for the linkage in the Common Heap, then it must fill in the details, then link the Thing in. Sadly, this requirement almost inevitably implies that you end up with two copies of the linkage block, the real one and the copy in your program code. Still, it makes sense. See the example given in QRM section 17!

Volume 16 and DVD

Right, here we are, at the end of Volume 15 and ready to start a new Volume 16 soon.

This issue will not be shipped from Austria, and I still did not manage to squeeze everything into the 32 pages to stay under 100g total. It's on me again ... treat it as a "THANK YOU" for all the early renewals. Seeing all the renewals coming in so early shows your support ... I don't think I've ever had such a high percentage in the past.

All the German automatic renewals already went to the bank middle of June.

To make life easier, I will add a note about whether I think you have renewed or not on the address sheet. If the sheet says you have not renewed yet and you think I am wrong, please contact me:

SMSQ@J-M-S.com

There are so many ways of sending money to me, that it is possible that I don't notice a payment here or there, especially, if the renewal form has not been returned and/or no email has been written to me.

Thanks to Rainer Wolkwitz, we have all issues excluding the one you are now reading in PDF format. I am sure we will have the complete set very soon after Rainer receives his copy - he was extremely quick in converting the large pile of issue masters.

The next issue will be shipped from Austria middle of September, as usual. I have already bought printable DVDs ... so you can look forward to receiving the DVD containing all the issues of Volume 1 to 15 of QL Today, just like Geoff wrote in his Editorial.

Although I have DVD burners in most of my computers, I have never ever "produced" a DVD with viewable contents myself in the past.

I can put all the PDFs onto the DVD like I do for backing up data, but I feel it deserves something nicer. I own Nero 7 Premium, so the tool should be here. If somebody has something better, easier to use, freeware ... any suggestion and help would be most welcome.

Also, I don't know which format to use. I don't even know if the printable DVDs + or - ... but I guess this should not matter nowadays, where all the DVD-ROMs and DVD-burners are "super multi" whatever. But again: suggestions and informative help is welcome.

So let me finish and repeat what I initially wrote: thanks a lot for you support - I am looking forward to the next Volume and I hope (there is still hope, as you can see) that I can become more active again in the QL scene ... not "just" by producing QL Today for you.

Latest News Digest

Shortly before QL Today was due to go to press several important news items came in. Here we present a short summary of these:

Rich Mellor of RWAP software has announced his support for the SER-USB interface in a recent email to customers and contacts.

To optimise use of the interface he has a limited quantity of Hermes, SuperHermes and Minerva chips for sale.

He is also selling QL keyboard membranes at a special offer price of £12 plus post and packing until the end of July.

For collectors he had the unique offer of what he describes as *"the original and rare Sinclair QL with Kludge"*

adding

"This extremely rare Sinclair QL is still available for purchase - boxed and fully working, this shows how the Sinclair QL was first released to the expectant market. Complete with the classic Kludge, containing part of the operating system, this contains some of the best microdrive units we have ever used (they format a cartridge in a few seconds)"

Earlier he had also announced a re-release of an F1 motor racing game, QL Vroom, in agreement with Pyramide's former owner.

The game costs £5.
http://www.rwapsoftware.co.uk
http://sellmyretro.com

Marcos Cruz announced he had finished a port from ZX Spectrum of a text adventure in Spanish. The details are in The Sinclair QL Forum:
http://qlforum.co.uk/viewtopic.php?f=3&t=163

Quanta has a new editor for its magazine. The current issue is being prepared by the Chairwoman, Sarah Gilpin, but the next will be the responsibility of Lee Privett. QL Today understands John Gilpin, the former acting editor of the Quanta Magazine, is a preparing detailed written guidance for Lee and other future editors of the magazine.

Lee is a former QL-er who returned to the scene last year. At the recent Quanta AGM he told the story of his decision to rejoin Quanta. While researching developments in the QL scene he visited the Quanta website and saw his own name in an image of the library guide.

Quanta is still experiencing problems with its website. As we go to press a notice on the website says that it is still undergoing maintenance. As an interim measure some Quanta information is available on Dilwyn Jones' website.

http://www.dilwyn.me.uk/gen/quanta/quanta.html

Meanwhile QL Today hears from the horse's mouth (or is it the bird's mouth) that **Dilwyn** now goes "Tweet, Tweet". He writes:

"Recent visitors to my website may have noticed that a couple of Twitter buttons have appeared on my website. Tweeting is something new for me, but I hope to use it to announce news of anything added to my website.

Clicking on the Twitter button takes you to a list of my Tweets (that word is already driving me mad) and there is a "tweet" button as well for anyone else on twitter to retweet or whatever. Not quite sure how it'll work or if I've set it up right yet, but hopefully once working properly will be another way to spread the word of the QL.

If anyone else on Twitter around here wants to send me private messages over it, send it to **@DilwynJones2**."

And finally, please remember to renew your QL Today subscription (if you have not done it yet)! The next issue will come with a free DVD for all subscribers!